Verilog 开发 MIPS 流水线处理器

一. 整体结构：

流水线处理器包括流水寄存器、各级组合逻辑以及各级控制器三大部分

它们均放在 mips.v 层次下，其中 code.txt 中存储相应指令码 处理器为 32 位处理器，支持的指令集为：addu,subu, ori, lw, sw, beq, lui, j,jal, jr,nop。

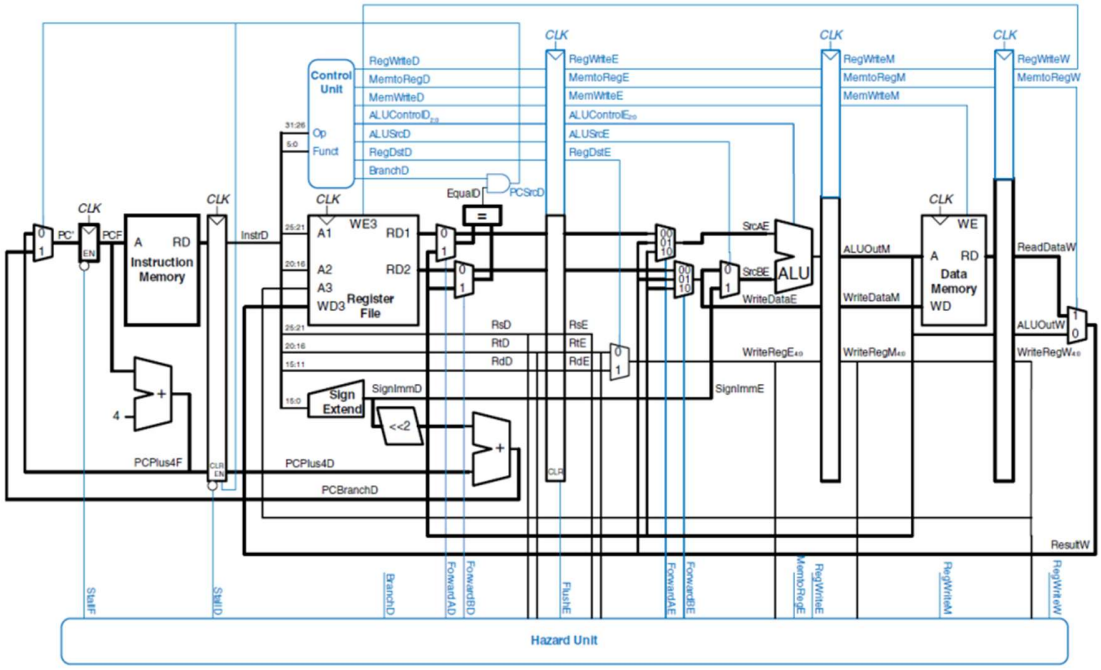


Figure 7.58 Pipelined processor with full hazard handling

二．数据通路部分

```
module mips(
input clk, //
input reset
);
    parameter delay_slot=1'b1;
    parameter condition=1'b1;//bxxal

    wire stall;
    wire [1:0]ForwardrsD;
    wire [1:0]ForwardrtD;
    wire [1:0]ForwardrsE;
    wire [1:0]ForwardrtE;
    wire ForwardrtM;

    wire [31:0]nextPC;
    wire [31:0]IAddrF;
    wire [31:0]IAddrD;
    wire [31:0]IAddrE;
    wire [31:0]IAddrM;
    wire [31:0]IAddrW;

    wire [31:0]InstrF;
    wire [31:0]InstrD;
    wire [31:0]InstrE;
    wire [31:0]InstrM;
    wire [31:0]InstrW;
    wire [31:0]PC4D;

    wire JumpD;
    wire [2:0]RegSrcD;
    wire MemWriteD;
    wire BranchD;
    wire [1:0]ALUSrcD;
    wire [1:0]RegDstD;
    wire RegWriteD;
    wire [1:0]ExtOpD;
    wire [3:0]ALUCtrlD;
    wire loenD;
    wire hienD;
```

```
43      wire JumpE;
44      wire [2:0]RegSrcE;
45      wire MemWriteE;
46      wire BranchE;
47      wire [1:0]ALUSrcE;
48      wire [1:0]RegDstE;
49      wire RegWriteE;
50      wire [1:0]ExtOpE;
51      wire [3:0]ALUCtrlE;
52      wire loenE;
53      wire hienE;
54
55      wire JumpM;
56      wire [2:0]RegSrcM;
57      wire MemWriteM;
58      wire BranchM;
59      wire [1:0]ALUSrcM;
60      wire [1:0]RegDstM;
61      wire RegWriteM;
62      wire [1:0]ExtOpM;
63      wire [3:0]ALUCtrlM;
64      wire loenM;
65      wire hienM;
66
67      wire JumpW;
68      wire [2:0]RegSrcW;
69      wire MemWriteW;
70      wire BranchW;
71      wire [1:0]ALUSrcW;
72      wire [1:0]RegDstW;
73      wire RegWriteW;
74      wire [1:0]ExtOpW;
75      wire [3:0]ALUCtrlW;
76      wire loenW;
77      wire hienW;
78
79      wire PCSrc;
80      //fetch
81      PC pc(clk,~stall,reset,nextPC,IAddrF);
82      Instr_Memory im(IAddrF[13:2]-12'hc00,InstrF);
83
84      preg32 InstrFD(clk,~stall,(~delay_slot&PCSrc&~stall)|reset,InstrF,InstrD);
85      preg32 PC4FD(clk,~stall,(~delay_slot&PCSrc&~stall)|reset,IAddrF+4,PC4D);
86      preg32 IAddrFD(clk,~stall,(~delay_slot&PCSrc&~stall)|reset,IAddrF,IAddrD);
```

```
87      //decode
88      Controller ctrlD(InstrD,JumpD,RegSrcD,MemWriteD,BranchD,ALUSrcD,RegDstD,RegWriteD,ExtOpD,ALUCtrlD,loenD,hienD);
89      wire [31:0]RegWDataM;
90      wire [31:0]RegWDataW;
91      wire [4:0]WriteRegM;
92      wire [4:0]WriteRegW;
93      wire trueW;
94      wire [4:0]rsD;
95      wire [4:0]rtD;
96      wire [4:0]rdD;
97      wire [31:0]RegRData1D;
98      wire [31:0]RegRData2D;
99      wire [31:0]ImmD;
00      wire [31:0]PC4E;
01      assign rsD=InstrD[25:21];
02      assign rtD=InstrD[20:16];
03      assign rdD=InstrD[15:11];
04      GRF rf(clk,BranchW&&condition? RegWriteW&&trueW: RegWriteW,reset,rsD,rtD,WriteRegW,RegWDataW,IAddrW,RegRData1D,RegRData2D);
05      ext immext(InstrD[15:0],ExtOpD,ImmD);
06      wire [31:0]cmp1;
07      wire [31:0]cmp2;
08      wire true;
09      assign PCSrc=BranchD&&true;
10      wire [31:0]jumpto;
11      assign cmp1=ForwardrsD==3? PC4E: ForwardrsD==2? RegWDataM: ForwardrsD==1? RegWDataW: RegRData1D;
12      assign cmp2=ForwardrtD==3? PC4E: ForwardrtD==2? RegWDataM: ForwardrtD==1? RegWDataW: RegRData2D;
13      compare cmp(cmp1,cmp2,ALUCtrlD[2:0],true);
14      assign jumpto=ALUSrcD[0]?{IAddrF[31:28],InstrD[25:0],2'b00}:cmp1;
15      assign nextPC=JumpD ? jumpto : PCSrc? PC4D+ImmD :IAddrF+4;
16
17      wire [31:0]RegRData1E;
18      wire [31:0]RegRData2E;
19      wire [4:0]rsE;
20      wire [4:0]rtE;
21      wire [4:0]rdE;
22      wire [31:0]ImmE;
23      wire trueE;
```

```
125       preg32 InstrDE(clk,1'b1,stall|reset,InstrD,InstrE);
126       preg32 PC4DE(clk,1'b1,stall|reset,(delay_slot?PC4D+4:PC4D),PC4E);
127       preg32 IAddrDE(clk,1'b1,stall|reset,IAddrD,IAddrE);
128       preg32 RD1DE(clk,1'b1,stall|reset,cmp1,RegRData1E);
129       preg32 RD2DE(clk,1'b1,stall|reset,cmp2,RegRData2E);
130       preg5 rsDE(clk,1'b1,stall|reset,rsD,rsE);
131       preg5 rtDE(clk,1'b1,stall|reset,rtD,rtE);
132       preg5 rdDE(clk,1'b1,stall|reset,rdD,rdE);
133       preg32 immDE(clk,1'b1,stall|reset,ImmD,ImmE);
134       preg1 trueDE(clk,1'b1,stall|reset,true,trueE);
135
136       //execute
137       Controller ctrlE(InstrE,JumpE,RegSrcE,MemWriteE,BranchE,ALUSrcE,RegDstE,RegWriteE,ExtOpE,ALUCtrlE,loenE,hienE);
138       wire [4:0]WriteRegE;
139       assign WriteRegE=RegDstE==2? 31: RegDstE==1? rdE:rtE;
140       wire [31:0]regAE;
141       wire [31:0]regBE;
142       wire [31:0]SrcAE;
143       wire [31:0]SrcBE;
144       wire [31:0]ALUOutE;
145       //wire [31:0]hiE;
146       //wire [31:0]loE;
147       //wire busy;
148       assign regAE=ForwardrsE==2? RegWDataM: ForwardrsE==1? RegWDataW: RegRData1E;
149       assign regBE=ForwardrtE==2? RegWDataM: ForwardrtE==1? RegWDataW: RegRData2E;
150       assign SrcAE=ALUSrcE[1]?{27'b0,InstrE[10:6]}:regAE;
151       assign SrcBE=ALUSrcE[0]?ImmE:regBE;
152       ALU a(SrcAE,SrcBE,ALUCtrlE,ALUOutE);
153       //muldiv hilo(clk,loenE,hienE,reset,InstrE[1],InstrE[0],InstrE[30]&&~InstrE[1],regAE,regBE,busy,loE,hiE);
154       wire [31:0]MemWDataE;
155       assign MemWDataE=regBE;
156
157       wire [31:0]MemWDataM;
158       wire [31:0]WData;
159       wire [31:0]ALUOutM;
160       wire [31:0]PC4M;
161       wire [4:0]rtM;
162       //wire [31:0]loM;
163       //wire [31:0]hiM;
164       wire trueM;
165       assign RegWDataM=RegSrcM==2?PC4M: ALUOutM;
166       assign WData=ForwardrtM?RegWDataW:MemWDataM;
167       preg32 InstrEM(clk,1'b1,reset,InstrE,InstrM);
168       preg32 ALUOutEM(clk,1'b1,reset,ALUOutE,ALUOutM);
169       preg32 MemWDataEM(clk,1'b1,reset,MemWDataE,MemWDataM);
170       preg32 PC4EM(clk,1'b1,reset,PC4E,PC4M);
171       preg32 IAddrEM(clk,1'b1,reset,IAddrE,IAddrM);
```

```
preg5 WriteRegEM(clk,1'b1,reset,WriteRegE,WriteRegM);
preg5 rtEM(clk,1'b1,reset,rtE,rtM);
preg1 trueEM(clk,1'b1,reset,trueE,trueM);
//memory
wire [31:0]MemRDataM;
Controller ctrlM(InstrM,JumpM,RegSrcM,MemWriteM,BranchM,ALUSrcM,RegDstM,RegWriteM,ExtOpM,ALUCtrlM,loenM,hienM);
DM_8bit dm(clk,MemWriteM,reset,InstrM[28],InstrM[27:26],ALUOutM[13:0],WData,IAddrM,MemRDataM);

wire [31:0]ALUOutW;
wire [31:0]PC4W;
wire [31:0]MemRDataW;
//wire [31:0]hiW;
//wire [31:0]loW;
preg32 InstrMW(clk,1'b1,reset,InstrM,InstrW);
preg32 ALUOutMW(clk,1'b1,reset,ALUOutM,ALUOutW);
preg32 PC4MW(clk,1'b1,reset,PC4M,PC4W);
preg32 IAddrMW(clk,1'b1,reset,IAddrM,IAddrW);
preg32 MemRDataMW(clk,1'b1,reset,MemRDataM,MemRDataW);
//preg32 hiMW(clk,1'b1,reset,hiM,hiW);
//preg32 loMW(clk,1'b1,reset,loM,loW);
preg5 WriteRegMW(clk,1'b1,reset,WriteRegM,WriteRegW);
preg1 trueMW(clk,1'b1,reset,trueM,trueW);
Controller ctrlW(InstrW,JumpW,RegSrcW,MemWriteW,BranchW,ALUSrcW,RegDstW,RegWriteW,ExtOpW,ALUCtrlW,loenW,hienW);
assign RegWDataW=RegSrcW==2? PC4W: RegSrcW==1? MemRDataW: ALUOutW;
```

```
    conflict hazard
    (WriteRegE,
    WriteRegM,
    BranchE&&condition? RegWriteE&&trueE: RegWriteE,
    BranchM&&condition? RegWriteM&&trueM: RegWriteM,
    RegSrcE,
    RegSrcM,
    BranchD,
    JumpD,
    JumpE,
    RegDstD,
    rsD,
    rtD,
    rsE,
    rtE,
    WriteRegW,
    BranchW&&condition? RegWriteW&&trueW: RegWriteW,
    rtM,
    stall,
    ForwardrsD,
    ForwardrtD,
    ForwardrsE,
    ForwardrtE,
    ForwardrtM);
endmodule
```

1. PC（PC.V）

初始值：0x30000000

```
module PC(
input clk, //时钟
input en,
input reset,
input [31:0]next,
output reg [31:0]IAddr=32'h00003000
);
```

端口定义:

信号名	方向	描述
next[31:0]	I	下一条指令地址
clk	I	时钟信号
reset	I	复位信号 1：有效 0：无效
reg[31:0]	O	当前指令地址 (IAddr=32'h00003000)

2．IM（im.v）

容量：32bit*1024 字，地址 10 位。只读，不可写。

端口定义：

信号名	方向	描述
RAddr[9:0]	I	指令地址后 10 位
RData[31:0]	O	指令机器码

```
module Instr_Memory(
input [9:0]RAddr,//指令地址后 10 位
output [31:0]RData//指令机器码
);
```

存储部件：`reg [31:0] rom[0:1023];`

初始化：`$readmemh("code.txt",rom);`

3．GRF（GRF.v）

具有写使能的寄存器实现，寄存器总数为 32 个 0 号寄存器的值始终保持为 0。其他寄存器初始值均为 0

端口定义：

信号名	方向	描述
IAddr[31:0]	I	相应指令存储地址
clk	I	时钟信号

reset	I	复位信号 1：有效 0：无效
WEnable	I	读写控制信号 1：写操作 0：读操作
RAddr1[4:0]	I	读寄存器 1 的地址
RAddr2[4:0]	I	读寄存器 2 的地址
WAddr[4:0]	I	5 为地址输入信号，指定 32 个寄存器中的一个作为写入目标寄存器地址（写地址）
WData[31:0]	I	向写寄存器中写入的值（写数据）
RData1[31:0]	O	32 位输出 1（读数据 1）
RData2[31:0]	O	32 位输出 2（读数据 2）

```
module GRF(
input clk,//时钟信号
input WEnable,//写使能
input reset,//同步复位
input [4:0]RAddr1,//读地址 1
input [4:0]RAddr2,//读地址 2
```

```
input [4:0]WAddr,//写地址
input [31:0]WData,//写数据
input [31:0]IAddr,//当前指令地址，仅用于控制台输出
output [31:0]RData1,//读数据 1
output [31:0]RData2//读数据 2
);
```

4．ALU（ALU.v）

端口定义：

信号名	方向	描述
op1[31:0]	I	ALU32 位输入数据 A（操作数 1）
op2[31:0]	I	ALU32 位输入数据 B（操作数 2）
sel[3:0]	I	ALU 功能选择信号
Result[31:0]	O	32 位数据输出（计算结果）
Zero	O	输出为 0

```
module ALU(
```

```
input [31:0]op1,//操作数 1
input [31:0]op2,//操作数 2
input [3:0]sel,//功能选择
output [31:0]result,//计算结果
output zero//计算结果为 0 标志位
);
```

选择信号：

0000---非负　0001---负数　0010---加法　0011---减法

0100---按位与　0101---按位或　0110---按位异或 0111---按位或非

1000---逻辑右移　1001---算术右移　1010---左移　1011---相等

1100---有符号小于 1101---无符号小于 1110---正数　1111---≤0

5．ext（ext.v）

容量：32bit*1024 字，地址 10 位

端口定义：

信号名	方向	描述
imm[15:0]	I	16 位 imm 数据输入

EOp[1:0]	I	位扩展选择信号 00: 符号扩展 01：无符号扩展 10：加载至高 16 位(lui) 11：符号扩展之后，左移两位
ext[31:0]	O	位扩展后的 32 位输出

```
module ext(
input [15:0]imm,//待扩展 16 位数
input [1:0]EOp,//扩展方式
output [31:0]ext//扩展后的 32 位数
);
```

扩展方式：

00：符号扩展

01：无符号扩展

10：加载至高 16 位(lui)

11：符号扩展之后，左移两位

6．DM（DM_8bit.v）

容量：32bit*1024 字。地址 10 位，可读可写可复位（同步复位）。

端口定义：

信号名	方向	描述
clk	I	时钟信号
WE	I	读写控制信号 1：写操作 0：读操作
reset	I	复位信号 1：有效 0：无效
isu	I	判断是否无符号或有符号数
MemDst[1:0]	I	写入数据的输入
Addr[11:0]	I	数据地址
WData[31:0]	I	写数据
IAddr[31:0]	I	指令地址，仅用于控制台输出
RData[31:0]	O	读数据

```
module Data_Memory(
input clk,//时钟信号
input WE,
input reset,//同步复位
input [9:0]Addr,//数据地址
input [31:0]WData,//写数据
```

```
input [31:0]IAddr,//指令地址，仅用于控制台输出
output [31:0]RData//读数据
);
```

存储部件：`reg [31:0] ram[0:1023];`

7．Controller(Controller.v)

端口定义：

信号名	方向	描述
cmd[31:0]	I	32 位指令码
Jump	O	跳转信号 0 为不是跳转指令 1 为是跳转指令
RegSrc[1:0]	O	寄存器数据来源
MemWrite	O	DM 写控制信号，写入 GRF 的数据选择(内存写使能信号)
Branch	O	分支信号 输出 0 为不是 Branch 输出为 1 是 Branch
ALUSrc[1:0]	O	ALU 操作数 2 的来源

RegDst[1:0]	O	寄存器地址选择 0:[20:16] 1:[15:11]
RegWrite	O	寄存器写使能信号
ExtOp[1:0]	O	控制位扩展方式
ALUCtrl[3:0]	O	ALU 功能选择信号

```
module Controller(
input [31:0]cmd,      //指令码
output Jump,          //跳转信号
output [1:0]RegSrc,  //寄存器数据来源
output MemWrite,     //写内存信号
output Branch,       //分支信号
output ALUSrc,       //ALU 操作数 2 来源
output [1:0]RegDst, //寄存器写地址选择
output RegWrite,    //写寄存器信号
output [1:0]ExtOp,  //位扩展方式
output [3:0]ALUCtrl //ALU 运算选择
);
```

详细解释：

RegSrc：00：ALU 01：DM 10：PC+4

RegDst：00：Instr[20:16] 01：Instr[15:11] 10：31（$ra）

8．Pipeline (pipelineregs.v)

```
module preg1(
input clk,
input en,
input reset,
input in,
output reg out=0
);

module preg5(
input clk,
input en,
input reset,
input [4:0]in,
output reg [4:0]out=0
);

module preg32(
input clk,
input en,
input reset,
```

```
input [31:0]in,
output reg [31:0]out=0
);
```

9．Compare (compare.v)

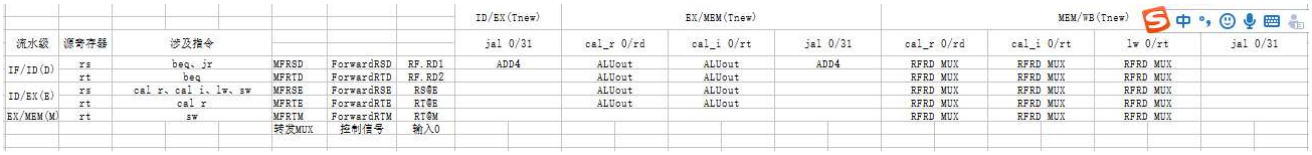
```
module compare(
input [31:0]c1,
input [31:0]c2,
input [2:0]sel,
output reg true
);
```

10．Conflict (conflict.v)

```
module conflict(
//stall
input [4:0]WriteRegE,
input [4:0]WriteRegM,
input RegWriteE,
input RegWriteM,
input [2:0]RegSrcE,
input [2:0]RegSrcM,
input BranchD,
input JumpD,
input JumpE,
input [1:0]RegDstD,
```

```
input [4:0]rsD,
input [4:0]rtD,

//forward
input [4:0]rsE,
input [4:0]rtE,
input [4:0]WriteRegW,
input RegWriteW,
input [4:0]rtM,

//input mdbusy,
//input usehilo,

//stall out
output stall,

//forward out
output [1:0]ForwardrsD,
output [1:0]ForwardrtD,
output [1:0]ForwardrsE,
output [1:0]ForwardrtE,
output ForwardrtM
);
```

三．转发和暂停



						ID/EX(Tnew)	EX/MEM(Tnew)			MEM/WB(Tnew)			
流水级	源寄存器	涉及指令				jal 0/31	cal_r 0/rd	cal_i 0/rt	jal 0/31	cal_r 0/rd	cal_i 0/rt	lw 0/rt	jal 0/31
IF/ID(D)	rs	beq、jr	MFRSD	ForwardRSD	RF.RD1	ADD4	ALUout	ALUout	ADD4	RFRD MUX	RFRD MUX	RFRD MUX	
	rt	beq	MFRTD	ForwardRTD	RF.RD2		ALUout	ALUout		RFRD MUX	RFRD MUX	RFRD MUX	
ID/EX(E)	rs	cal_r、cal_i、lw、sw	MFRSE	ForwardRSE	RS@E		ALUout	ALUout		RFRD MUX	RFRD MUX	RFRD MUX	
	rt	cal_r	MFRTE	ForwardRTE	RT@E		ALUout	ALUout		RFRD MUX	RFRD MUX	RFRD MUX	
EX/MEM(M	rt	sw	MFRTM	ForwardRTM	RT@M					RFRD MUX	RFRD MUX	RFRD MUX	
			转发MUX	控制信号	输入0								

IF/ID当前指令			ID/EX(Tnew)			EX/MEM(Tnew)
指令类型	源寄存器	Tuse	cal_r 1/rd	cal_i 1/rt	lw 2/rt	lw 1/rt
cal_r	rs/rt	1			暂停	
cal_i	rs	1			暂停	
lw	rs	1			暂停	
sw	rs	1			暂停	
	rt	2				
beq	rs/rt	0	暂停	暂停	暂停	暂停
jr	rs	0	暂停	暂停	暂停	暂停
jalr	rs	0	暂停	暂停	暂停	暂停

对于暂停信号 stall

stall_B_Calr=(B_D&&Cal_r_E&&((IR_D_out[25:21]==IR_E_out[15:11])|(IR_D_out[20:16]==IR_E_out[15:11])));

stall_B_Cali=(B_D&&Cal_i_E&&((IR_D_out[25:21]==IR_E_out[20:16])|(IR_D_out[20:16]==IR_E_out[20:16])));

stall_B_Load1=(B_D&&Load_E&&((IR_D_out[25:21]==IR_E_out[20:16])|(IR_D_out[20:16]==IR_E_out[20:16])));

stall_B_Load2=(B_D&&Load_M&&((IR_D_out[25:21]==IR_M_out[20:16])|(IR_D_out[20:16]==IR_M_out[20:16])));

stall_Calr_Load=(Cal_r_D&&Load_E&&((IR_D_out[25:21]==IR_E_out[20:16])|(IR_D_out[20:16]==IR_E_out[20:16])));

stall_Cali_Load=(Cal_i_D&&Load_E&&(IR_D_out[25:21]==IR_E_out[20:16]));

stall_Load_Load =(Load_D&&Load_E&&(IR_D_out[25:21]==IR_E_out[20:16]));

stall_Save_Load =(Save_D&&Load_E&&(IR_D_out[25:21]==IR_E_out[20:16]));

stall=stall_B_Calr|stall_B_Cali|stall_B_Load1|stall_B_Load2|stall_Calr_Load|stall_Cali_Load|stall_Load_Load|stall_Save_Load;

四．思考题

1. 在本实验中你遇到了哪些不同指令组合产生的冲突？你又是如何解决的？相应的测试样例是什么样的？请有条理的罗列出来。(非常重要)

（1）lw 后加 beq 指令出现异常，lw 后加 beq 应该暂停两次。

测试程序:

```
ori $a0,$0,2026
sw $a0,0($0)
lw $a1,0($0)
lw $a2,0($0)
beq $a2,$a1,branch
ori $t0,$0,1
ori $t0,$0,2
branch:
ori $t1,$0,3
```

预期结果:

```
 35@00003000: $ 4 <= 000007ea
 35@00003004: *00000000 <= 000007ea
 55@00003008: $ 5 <= 000007ea
 65@0000300c: $ 6 <= 000007ea
105@00003014: $ 8 <= 00000001
115@0000301c: $ 9 <= 00000003
```

（2）jr 在 D 级需要用到 rs 寄存器的值，需要通过转发来解决，不然中强测前两个点过不去。

测试程序:

```
ori $ra,$0,0x3014
jr $ra
ori $a0,$0,1
ori $a0,$0,2
ori $a0,$0,3
ori $a0,$0,4
```

预期结果：

```
35@00003000: $31 <= 00003014
65@00003008: $ 4 <= 00000001
75@00003014: $ 4 <= 00000004
```

（3）在 D 级部件将通过转发后的 RF_RD1_trans 和 RF_RD2_trans 转发到了下一级寄存器中，实际上应该将转发前的 RF_RD1 和 RF_RD2 传到下一级。

（4）jal 延迟槽后跟 jr 发生冲突，在 E 级增加多路选择器，，选择传出的地址是 ALU 计算出来的值还是 PC+8

测试程序：

```
jal jump
nop
jump:
jr $ra
ori $a0,$0,1
```

预期结果：

```
    35@00003000: $31 <= 00003008
    65@0000300c: $ 4 <= 00000001
    85@0000300c: $ 4 <= 00000001
   105@0000300c: $ 4 <= 00000001
   125@0000300c: $ 4 <= 00000001
   145@0000300c: $ 4 <= 00000001
   165@0000300c: $ 4 <= 00000001
   185@0000300c: $ 4 <= 00000001
   205@0000300c: $ 4 <= 00000001
   225@0000300c: $ 4 <= 00000001
   245@0000300c: $ 4 <= 00000001
   265@0000300c: $ 4 <= 00000001
   285@0000300c: $ 4 <= 00000001
   305@0000300c: $ 4 <= 00000001
   325@0000300c: $ 4 <= 00000001
   345@0000300c: $ 4 <= 00000001
   365@0000300c: $ 4 <= 00000001
   385@0000300c: $ 4 <= 00000001
   405@0000300c: $ 4 <= 00000001
   425@0000300c: $ 4 <= 00000001
   445@0000300c: $ 4 <= 00000001
   465@0000300c: $ 4 <= 00000001
   485@0000300c: $ 4 <= 00000001
   505@0000300c: $ 4 <= 00000001
   525@0000300c: $ 4 <= 00000001
   545@0000300c: $ 4 <= 00000001
   565@0000300c: $ 4 <= 00000001
   585@0000300c: $ 4 <= 00000001
   605@0000300c: $ 4 <= 00000001
   625@0000300c: $ 4 <= 00000001
   645@0000300c: $ 4 <= 00000001
   665@0000300c: $ 4 <= 00000001
```

（5）beq 指令当跳转条件不满足时，应传回 PC+8 的值，而不是 PC+4，不然中强测第二个点过不去。

测试程序:

```
ori $a0,$0,1
ori $a1,$0,2
beq $a0,$a1,jump
ori $t0,$0,3
ori $t1,$0,4
addu $t7,$a0,$a1
jump:
subu $t7,$a1,$a0
```

预期结果:

```
 35@00003000: $ 4 <= 00000001
 45@00003004: $ 5 <= 00000002
 75@0000300c: $ 8 <= 00000003
 85@00003010: $ 9 <= 00000004
 95@00003014: $15 <= 00000003
105@00003018: $15 <= 00000001
```

五．测试程序

（一）整体测试

```
1  ori $t0,$0,2       #t0---2
2  ori $t1,$0,4       #t1---4
3  beq $t0,$t1,jump1  #不跳
4  addu $a0,$t0,$t1   #a0---6
5  subu $a0,$t1,$t0         #a0---2
6  jump1:
7  ori $t3,$0,3     #t3---3
8  ori $a1,$0,2     #a1---2
9  beq $a0,$a1,jump2  #跳
10 ori $t7,$0,8
11 ori $t7,$0,9
12 jump2:
13 ori $ra,$0,0x303c
14 jr $ra
15 ori $a0,$0,1
16 ori $a0,$0,2
```

```
16  ori $a0,$0,2
17  ori $a0,$0,3
18  ori $a0,$0,4      #跳到这步
19  ori $a0,$0,2026
20  sw $a0,0($0)
21  lw $a1,0($0)
22  lw $a2,0($0)
23  beq $a1,$a2,branch
24  ori $t0,$0,1
25  ori $t1,$0,2
26  branch:
27  ori $t1,$0,3
28  jal jump
29  lui $s1,1
30  lui $s2,2
31  jump:
```

```
32  lui $s3,3
33  j finish
34  finish:
35  jr $ra
```

预期结果:

```
 35@00003000: $ 8 <= 00000002
 45@00003004: $ 9 <= 00000004
 75@0000300c: $ 4 <= 00000006
 85@00003010: $ 4 <= 00000002
 95@00003014: $11 <= 00000003
105@00003018: $ 5 <= 00000002
135@00003020: $15 <= 00000008
145@00003028: $31 <= 0000303c
175@00003030: $ 4 <= 00000001
185@0000303c: $ 4 <= 00000004
195@00003040: $ 4 <= 000007ea
195@00003044: *00000000 <= 000007ea
215@00003048: $ 5 <= 000007ea
225@0000304c: $ 6 <= 000007ea
265@00003054: $ 8 <= 00000001
275@0000305c: $ 9 <= 00000003
285@00003060: $31 <= 00003068
295@00003064: $17 <= 00010000
305@0000306c: $19 <= 00030000
345@00003068: $18 <= 00020000
355@00003068: $18 <= 00020000
365@0000306c: $19 <= 00030000
405@00003068: $18 <= 00020000
415@00003068: $18 <= 00020000
425@0000306c: $19 <= 00030000
465@00003068: $18 <= 00020000
475@00003068: $18 <= 00020000
485@0000306c: $19 <= 00030000
525@00003068: $18 <= 00020000
535@00003068: $18 <= 00020000
545@0000306c: $19 <= 00030000
585@00003068: $18 <= 00020000
595@00003068: $18 <= 00020000
605@0000306c: $19 <= 00030000
```

（二）冲突测试

对于冲突的测试类型可以用：X - Y - Z 来表示，它们的含义如下。

- X：产生冲突的前序指令的类型。
- Y：前序指令在哪个阶段与当前指令产生冲突。
- Z：产生冲突的寄存器。

如：

用例编号	测试类型	前序指令	冲突位置	冲突寄存器	测试序列
1	R-M-RS	subu	MEM	rs	subu $1, $2, $3 addu $4, $1, $2
2	R-M-RT	subu	MEM	rt	subu $1, $2, $3 addu $4, $2, $1

3	R-W-RS	subu	MEM	rs	subu $1, $2, $3
					instru 无关
					addu $4, $1, $2

4	R-M-RT	subu	MEM	rt	subu $1, $2, $3
					instru 无关
					addu $4, $2, $1

5	I-M-RS	ori	MEM	rs	ori $1, $2, 1000 addu $4, $1, $2
6	I-M-RT	ori	MEM	rt	ori $1, $2, 1000 addu $4, $2, $1
7	I-W-RS	ori	MEM	rs	ori $1, $2, 1000 instru 无关 addu $4, $1, $2
8	I-W-RT	ori	MEM	rt	ori $1, $2, 1000 instru 无关 addu $4, $2, $1
9	LD-M-RS				
10	LD-M-RT				
11	LD-W-RS				
12	LD-W-RT				

（1）Cal_r 类型指令

1) R—M—RS

测试序列：　　预期结果：

```
ori $t0, $0, 1
ori $t1, $0, 2
ori $t3, $0, 6
subu $s1, $t3, $t0
addu $s2, $s1, $t3
```

```
35@00003000: $ 8 <= 00000001
45@00003004: $ 9 <= 00000002
55@00003008: $11 <= 00000006
65@0000300c: $17 <= 00000005
75@00003010: $18 <= 0000000b
```

2）R—M—RT

测试序列：　　预期结果：

```
ori $t0, $0, 1
ori $t1, $0, 2
ori $t3, $0, 6
subu $s1, $t3, $t0
addu $s2, $t3, $s1
```

```
35@00003000: $ 8 <= 00000001
45@00003004: $ 9 <= 00000002
55@00003008: $11 <= 00000006
65@0000300c: $17 <= 00000005
75@00003010: $18 <= 0000000b
```

3）R—W—RS　　预期结果：

```
ori $t0, $0, 1
ori $t1, $0, 2
ori $t3, $0, 6
subu $s1, $t3, $t0
ori $t4, $0, 3
addu $s2, $s1, $t3
```

```
35@00003000: $ 8 <= 00000001
45@00003004: $ 9 <= 00000002
55@00003008: $11 <= 00000006
65@0000300c: $17 <= 00000005
75@00003010: $12 <= 00000003
85@00003014: $18 <= 0000000b
```

4）R—W—RT　　预期结果：

```
ori $t0, $0, 1
ori $t1, $0, 2
ori $t3, $0, 6
subu $s1, $t3, $t0
ori $t4, $0, 3
addu $s2, $t3, $s1
```

```
35@00003000: $ 8 <= 00000001
45@00003004: $ 9 <= 00000002
55@00003008: $11 <= 00000006
65@0000300c: $17 <= 00000005
75@00003010: $12 <= 00000003
85@00003014: $18 <= 0000000b
```

5）I—M—RS　　预期结果：

```
ori $t1, $0, 2
ori $t0, $0, 10
addu $s1, $t0, $t1
```

```
35@00003000: $ 9 <= 00000002
45@00003004: $ 8 <= 0000000a
55@00003008: $17 <= 0000000c
```

6）I—M—RT　　预期结果：

```
ori $t1, $0, 2
ori $t0, $0, 10
addu $s1, $t1, $t0
```

```
35@00003000: $ 9 <= 00000002
45@00003004: $ 8 <= 0000000a
55@00003008: $17 <= 0000000c
```

7）I—W—RS

```
ori $t1, $0, 2
ori $t0, $0, 10
ori $t4, $0, 5
addu $s1, $t0, $t1
```

```
35@00003000: $ 9 <= 00000002
45@00003004: $ 8 <= 0000000a
55@00003008: $12 <= 00000005
65@0000300c: $17 <= 0000000c
```

8）I—W—RT　　预期结果：

```
ori $t1, $0, 2
ori $t0, $0, 10
ori $t4, $0, 5
addu $s1, $t1, $t0
```

```
35@00003000: $ 9 <= 00000002
45@00003004: $ 8 <= 0000000a
55@00003008: $12 <= 00000005
65@0000300c: $17 <= 0000000c
```

9）LD—M—RS　　预期结果：

```
ori $t0, $0, 10
sw $t0, 0($sp)
lw $t1, 0($sp)
addu $s1, $t1, $t0
```

```
35@00003000: $ 8 <= 0000000a
35@00003004: *00000000 <= 0000000a
55@00003008: $ 9 <= 0000000a
75@0000300c: $17 <= 00000014
```

10）LD—M—RT　　预期结果：

```
ori $t0, $0, 10
sw $t0, 0($sp)
lw $t1, 0($sp)
addu $s1, $t0, $t1
```

```
35@00003000: $ 8 <= 0000000a
35@00003004: *00000000 <= 0000000a
55@00003008: $ 9 <= 0000000a
75@0000300c: $17 <= 00000014
```

11）LD—W—RS　　预期结果：

```
ori $t0, $0, 10
sw $t0, 0($sp)
lw $t1, 0($sp)
ori $t3, $0, 5
addu $s1, $t1, $t0
```

```
35@00003000: $ 8 <= 0000000a
35@00003004: *00000000 <= 0000000a
55@00003008: $ 9 <= 0000000a
65@0000300c: $11 <= 00000005
75@00003010: $17 <= 00000014
```

12）LD—W—RT　　预期结果：

```
ori $t0, $0, 10
sw $t0, 0($sp)
lw $t1, 0($sp)
ori $t3, $0, 5
addu $s1, $t0, $t1
```

```
35@00003000: $ 8 <= 0000000a
35@00003004: *00000000 <= 0000000a
55@00003008: $ 9 <= 0000000a
65@0000300c: $11 <= 00000005
75@00003010: $17 <= 00000014
```

Cal_r 整体测试：

```
ori $0, 50

ori $1, 100

ori $2, 200

ori $3, 300

ori $4, 400

ori $5, 500

top:

beq $1, $4, top

nop

ori $6, 600

ori $7, 700

ori $8, 800

ori $9, 900

ori $10, 1000
```

```
ori $11, 100

ori $12, 200

ori $13, 300

ori $14, 100

ori $15, 200

ori $16, 300

ori $17, 50

ori $18, 100

ori $19, 200

ori $20, 300

ori $21, 0x0008

ori $22, 0x0048

ori $23, 0x2ffc

ori $24, 0x120

ori $25, 0x0004

#1

subu $1, $2, $3  # $1=-100

addu $4, $1, $2 # $4=100

#2

subu $1, $2, $3 # $1=-100

addu $4, $3, $1 # $4=200

#3

subu $11, $12, $13 # $11=-100
```

```
sw $4, -4($23)    # *00002ffc=200

addu $14, $11, $12 # $14=100

#4

subu $11, $12, $13 # $11=-100

subu $23, $23, $25 # $23=0x2ff8

addu $14, $12, $11 # $14=100

#8

ori $1, $0, 128   # $1=128

lui $16 1234

addu $4, $2, $1              # $4=228

#9

lw $1,0($23)

addu $4, $1, $2     # $4=400

#10

lw $1,0($23)

addu $4, $2, $1     # $4=400

#11

lw $1,0($23)

ori $27, 0x2ff4

addu $4, $1, $2     # $4=400

#12

lw $1,4($27)

subu $23, $23, $25
```

```
addu $4, $3, $1     # $4=600
#13
jal loop1
addu $4, $31, $1
ori $1, $4, 0
beq $1, $4, loop2
loop1:
#5
ori $1, $0, 128   # $1=128
addu $4, $1, $2             # $4=228
#6
ori $1, $0, 128   # $1=128
addu $4, $2, $1             # $4=228
#7
ori $2, $0, 128   # $1=128
lui $26, 8
addu $4, $2, $1             # $4=228
jr $31

loop2:
nop
jal loop3
addu $4, $1, $31
```

```
ori $1, $4, 0

beq $1, $4, loop4

loop3:

#5

ori $1, $0, 128   # $1=128

addu $4, $1, $2             # $4=228

#6

ori $1, $0, 128   # $1=128

addu $4, $2, $1             # $4=228

#7

ori $2, $0, 128   # $1=128

lui $26, 8

addu $4, $2, $1             # $4=228

jr $31


loop4:

sw $4, 0($23)

jal loop5

sw $4, 0($0)

ori $1, $4, 0

beq $1, $4, loop6

loop5:

addu $4, $31, $1
```

```
jr $31


loop6:

lw $5, 0($23)

jal loop8

ori $1, $4, 0

beq $1, $4, loop7

loop8:

addu $4, $1, $31

jr $31


loop7:

ori $1, $4, 0

beq $0, $0, top

lui $0, 100
```

预期输出:

```
45@00003004: $ 1 <= 00000064
55@00003008: $ 2 <= 000000c8
65@0000300c: $ 3 <= 0000012c
75@00003010: $ 4 <= 00000190
85@00003014: $ 5 <= 000001f4
115@00003020: $ 6 <= 00000258
125@00003024: $ 7 <= 000002bc
135@00003028: $ 8 <= 00000320
145@0000302c: $ 9 <= 00000384
155@00003030: $10 <= 000003e8
165@00003034: $11 <= 00000064
175@00003038: $12 <= 000000c8
185@0000303c: $13 <= 0000012c
```

195@00003040: $14 <= 00000064
205@00003044: $15 <= 000000c8
215@00003048: $16 <= 0000012c
225@0000304c: $17 <= 00000032
235@00003050: $18 <= 00000064
245@00003054: $19 <= 000000c8
255@00003058: $20 <= 0000012c
265@0000305c: $21 <= 00000008
275@00003060: $22 <= 00000048
285@00003064: $23 <= 00002ffc
295@00003068: $24 <= 00000120
305@0000306c: $25 <= 00000004
315@00003070: $ 1 <= ffffff9c
325@00003074: $ 4 <= 00000064
335@00003078: $ 1 <= ffffff9c
345@0000307c: $ 4 <= 000000c8
355@00003080: $11 <= ffffff9c
355@00003084: *00002ff8 <= 000000c8
375@00003088: $14 <= 00000064
385@0000308c: $11 <= ffffff9c
395@00003090: $23 <= 00002ff8
405@00003094: $14 <= 00000064
415@00003098: $ 1 <= 00000080
425@0000309c: $16 <= 04d20000
435@000030a0: $ 4 <= 00000148
445@000030a4: $ 1 <= 000000c8
465@000030a8: $ 4 <= 00000190
475@000030ac: $ 1 <= 000000c8
495@000030b0: $ 4 <= 00000190
505@000030b4: $ 1 <= 000000c8
515@000030b8: $27 <= 00002ff4
525@000030bc: $ 4 <= 00000190
535@000030c0: $ 1 <= 000000c8
545@000030c4: $23 <= 00002ff4
555@000030c8: $ 4 <= 000001f4
565@000030cc: $31 <= 000030d4
575@000030d0: $ 4 <= 0000319c
585@000030dc: $ 1 <= 00000080
595@000030e0: $ 4 <= 00000148
605@000030e4: $ 1 <= 00000080
615@000030e8: $ 4 <= 00000148
625@000030ec: $ 2 <= 00000080
635@000030f0: $26 <= 00080000

645@000030f4: $ 4 <= 00000100
675@000030d4: $ 1 <= 00000100
705@000030dc: $ 1 <= 00000080
725@00003100: $31 <= 00003108
735@00003104: $ 4 <= 00003188
745@00003110: $ 1 <= 00000080
755@00003114: $ 4 <= 00000100
765@00003118: $ 1 <= 00000080
775@0000311c: $ 4 <= 00000100
785@00003120: $ 2 <= 00000080
795@00003124: $26 <= 00080000
805@00003128: $ 4 <= 00000100
815@00003130: *00002ff4 <= 00000100
835@00003108: $ 1 <= 00000100
865@00003110: $ 1 <= 00000080
865@00003130: *00002ff4 <= 00000100
885@00003134: $31 <= 0000313c
885@00003138: *00000000 <= 00000100
905@00003144: $ 4 <= 000031bc
925@0000314c: $ 5 <= 00000100
935@0000313c: $ 1 <= 000031bc
965@00003144: $ 4 <= 000062f8
975@0000314c: $ 5 <= 00000100
985@00003150: $31 <= 00003158
995@00003154: $ 1 <= 000062f8
ISim>
run 1.00us
1005@0000315c: $ 4 <= 00009450
1025@00003164: $ 1 <= 00009450
1055@0000315c: $ 4 <= 0000c5a8
1065@00003164: $ 1 <= 0000c5a8
（2）Cal_i 整体测试

```
ori $t1,$0,7

ori $t2,$t1,8



ori $t1,$0,7

addu $s1,$0,$0

ori $t2,$t1,8
```

```
ori $t0,$0,3

ori $t1,$0,4

addu $t2,$t0,$t1

ori $t3,$t2,8


ori $t0,$0,7

sw $t0,0($sp)

lw $t1,0($sp)

ori $t2,$t1,8


jal jump

ori $t0,$0,1

jump:

ori $t1,$0,2
```

预期输出:

```
35@00003000: $ 9 <= 00000007
45@00003004: $10 <= 0000000f
55@00003008: $ 9 <= 00000007
65@0000300c: $17 <= 00000000
75@00003010: $10 <= 0000000f
85@00003014: $ 8 <= 00000003
95@00003018: $ 9 <= 00000004
105@0000301c: $10 <= 00000007
115@00003020: $11 <= 0000000f
125@00003024: $ 8 <= 00000007
125@00003028: *00000000 <= 00000007
145@0000302c: $ 9 <= 00000007
165@00003030: $10 <= 0000000f
```

175@00003034: $31 <= 0000303c
185@00003038: $ 8 <= 00000001
195@0000303c: $ 9 <= 00000002

P5 测试指令：

Addu:

Ori_E_RS(addu):

ori $t0, $zero, 8

addu $t1, $t0, $zero

180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000008

Ori_M_RS(addu)
ori $t0, $zero, 8
ori $t2, $zero, 12
addu $t1, $t0, $zero
180@00003000: $ 8 <= 00000008
220@00003004: $10 <= 0000000c
260@00003008: $ 9 <= 00000008

Ori_W_RS(addu)
ori $t0, $zero, 8
ori $t2, $zero, 12
ori $t3, $zero, 13
addu $t1, $t0, $zero
180@00003000: $ 8 <= 00000008
220@00003004: $10 <= 0000000c
260@00003008: $11 <= 0000000d
300@0000300c: $ 9 <= 00000008

Ori_E_RT(addu)
ori $t0, $zero, 8
addu $t1, $zero, $t0
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000008

Ori_M_RT(addu)
ori $t0, $zero, 8
ori $t2, $zero, 12
addu $t1, $zero, $t0
180@00003000: $ 8 <= 00000008
220@00003004: $10 <= 0000000c
260@00003008: $ 9 <= 00000008

Ori_W_RT(addu)
ori $t0, $zero, 8
ori $t2, $zero, 12
ori $t3, $zero, 13
addu $t1, $zero, $t0
180@00003000: $ 8 <= 00000008
220@00003004: $10 <= 0000000c
260@00003008: $11 <= 0000000d
300@0000300c: $ 9 <= 00000008

Lui_E_RS(addu)
lui $t0, 8
addu $t1, $t0, $zero
180@00003000: $ 8 <= 00080000
220@00003004: $ 9 <= 00080000

Lui_M_RS(addu)
lui $t0, 8
lui $t2, 12
addu $t1, $t0, $zero
180@00003000: $ 8 <= 00080000
220@00003004: $10 <= 000c0000
260@00003008: $ 9 <= 00080000

Lui_W_RS(addu)
lui $t0, 8
lui $t2, 12
lui $t3, 14
addu $t1, $t0, $zero
180@00003000: $ 8 <= 00080000
220@00003004: $10 <= 000c0000
260@00003008: $11 <= 000e0000
300@0000300c: $ 9 <= 00080000

Lui_W_RT(addu)
lui $t0, 8
addu $t1, $zero, $t0
180@00003000: $ 8 <= 00080000
220@00003004: $ 9 <= 00080000

Lui_w_RT(addu)
lui $t0, 8
lui $t2, 12
addu $t1, $zero, $t0
180@00003000: $ 8 <= 00080000
220@00003004: $10 <= 000c0000
260@00003008: $ 9 <= 00080000

Lui_W_RT(addu)
lui $t0, 8
lui $t2, 12
lui $t3, 14
addu $t1, $zero, $t0
180@00003000: $ 8 <= 00080000
220@00003004: $10 <= 000c0000
260@00003008: $11 <= 000e0000
300@0000300c: $ 9 <= 00080000

R_E_RS(addu)
lui $t0, 8
addu $t1, $zero, $t0
addu $t2, $t1, $zero
180@00003000: $ 8 <= 00080000
220@00003004: $ 9 <= 00080000
260@00003008: $10 <= 00080000

R_M_RS(addu)
lui $t0, 8
addu $t1, $zero, $t0
addu $t3, $t0, $t0
addu $t2, $t1, $zero
180@00003000: $ 8 <= 00080000
220@00003004: $ 9 <= 00080000
260@00003008: $11 <= 00100000
300@0000300c: $10 <= 00080000

R_W_RS(RT)(addu)
lui $t0, 8
addu $t1, $zero, $t0
addu $t3, $t0, $t0
addu $t4, $t0, $t0
addu $t2, $t1, $t1
180@00003000: $ 8 <= 00080000
220@00003004: $ 9 <= 00080000
260@00003008: $11 <= 00100000
300@0000300c: $12 <= 00100000
340@00003010: $10 <= 00100000

Ld_E_RS(RT)(addu)
ori $t0, $zero, 8
ori $t1, $zero, 12
ori $t2, $zero, 16
ori $t3, $zero, 20
ori $t4, $zero, 24
ori $t5, $zero, 28
sw $t1, 0($t0)
ori $s0, $zero, 4
ori $s1, $zero, 8
ori $s2, $zero, 12
lw $t6, 0($t0)
addu $t7, $t6, $t6
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 0000000c
260@00003008: $10 <= 00000010
300@0000300c: $11 <= 00000014
340@00003010: $12 <= 00000018
380@00003014: $13 <= 0000001c
380@00003018: *00000008 <= 0000000c

460@0000301c: $16 <= 00000004
500@00003020: $17 <= 00000008
540@00003024: $18 <= 0000000c
580@00003028: $14 <= 0000000c
660@0000302c: $15 <= 00000018

Ld_M_RS(RT)(addu)
ori $t0, $zero, 8
ori $t1, $zero, 12
ori $t2, $zero, 16
ori $t3, $zero, 20
ori $t4, $zero, 24
sw $t1, 0($t0)
ori $s0, $zero, 4
ori $s1, $zero, 8
ori $s2, $zero, 12
lw $t6, 0($t0)
ori $t5, $zero, 28
addu $t7, $t6, $t6
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 0000000c
260@00003008: $10 <= 00000010
300@0000300c: $11 <= 00000014
340@00003010: $12 <= 00000018
340@00003014: *00000008 <= 0000000c
420@00003018: $16 <= 00000004
460@0000301c: $17 <= 00000008
500@00003020: $18 <= 0000000c
540@00003024: $14 <= 0000000c
580@00003028: $13 <= 0000001c
620@0000302c: $15 <= 00000018

Ld_W_RS(RT)(addu)
ori $t0, $zero, 8
ori $t1, $zero, 12
ori $t2, $zero, 16
ori $t3, $zero, 20
ori $t4, $zero, 24
sw $t1, 0($t0)
ori $s0, $zero, 4
ori $s1, $zero, 8
ori $s2, $zero, 12
lw $t6, 0($t0)

```
ori $t5, $zero, 28
ori $t8, $zero, 32
addu $t7, $t6, $t6
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 0000000c
260@00003008: $10 <= 00000010
300@0000300c: $11 <= 00000014
340@00003010: $12 <= 00000018
340@00003014: *00000008 <= 0000000c
420@00003018: $16 <= 00000004
460@0000301c: $17 <= 00000008
500@00003020: $18 <= 0000000c
540@00003024: $14 <= 0000000c
580@00003028: $13 <= 0000001c
620@0000302c: $24 <= 00000020
660@00003030: $15 <= 00000018

Jal_E_RS(RT)(addu)
ori $t0, $zero, 8
ori $t1, $zero, 12
ori $t2, $zero, 16
jal change1
addu $t3, $ra, $ra
ori $t4, $zero, 20
ori $t5, $zero, 24
change1:
        ori $t6, $zero, 20
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 0000000c
260@00003008: $10 <= 00000010
300@0000300c: $31 <= 00003014
340@00003010: $11 <= 00006028
380@0000301c: $14 <= 00000014

Jal_M_RS(RT)(addu)
ori $t0, $zero, 8
ori $t1, $zero, 12
ori $t2, $zero, 16
jal change1
ori $t4, $zero, 20
ori $t5, $zero, 24
change1:
        addu $t3, $ra, $ra
```

```
        ori $t6, $zero, 20
        ori $t7, $zero, 24
$ 8 <= 00000008
$ 9 <= 0000000c
$10 <= 00000010
$31 <= 00003014
$12 <= 00000014
$11 <= 00006028
$14 <= 00000014
$15 <= 00000018


Jal_W_RS(RT)(addu)
ori $t0, $zero, 8
ori $t1, $zero, 12
ori $t2, $zero, 16
jal change1
ori $t4, $zero, 20
ori $t5, $zero, 24
change1:
        ori $t6, $zero, 20
        addu $t3, $ra, $ra
        ori $t6, $zero, 20
        ori $t7, $zero, 24
$ 8 <= 00000008
$ 9 <= 0000000c
$10 <= 00000010
$31 <= 00003014
$12 <= 00000014
$14 <= 00000014
$11 <= 00006028
$14 <= 00000014
$15 <= 00000018

Ori_E_RS(ori)
ori $t0, $zero, 8
ori $t1, $t0, 12
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 0000000c

Ori_M_RS(ori)
ori $t0, $zero, 8
ori $t2, $zero, 20
```

ori $t1, $t0, 12
180@00003000: $ 8 <= 00000008
220@00003004: $10 <= 00000014
260@00003008: $ 9 <= 0000000c

Ori_W_RS(ori)
ori $t0, $zero, 8
ori $t2, $zero, 20
ori $t3, $zero, 24
ori $t1, $t0, 12
180@00003000: $ 8 <= 00000008
220@00003004: $10 <= 00000014
260@00003008: $11 <= 00000018
300@0000300c: $ 9 <= 0000000c

Subu_E_RS(ori)
ori $t0, $zero, 8
ori $t1, $zero, 20
ori $t2, $zero, 24
ori $t3, $zero, 12
ori $t4, $zero, 16
subu $t5, $t0, $t1
ori $t6, $t5, 13
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000014
260@00003008: $10 <= 00000018
300@0000300c: $11 <= 0000000c
340@00003010: $12 <= 00000010
380@00003014: $13 <= fffffff4
420@00003018: $14 <= fffffffd

Subu_M_RS(ori)
ori $t0, $zero, 8
ori $t1, $zero, 20
ori $t2, $zero, 24
ori $t3, $zero, 12
ori $t4, $zero, 16
subu $t5, $t0, $t1
ori $t7, $zero, 20
ori $t6, $t5, 13
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000014
260@00003008: $10 <= 00000018

300@0000300c: $11 <= 0000000c
340@00003010: $12 <= 00000010
380@00003014: $13 <= fffffff4
420@00003018: $15 <= 00000014
460@0000301c: $14 <= fffffffd

Subu_W_RS(ori)
ori $t0, $zero, 8
ori $t1, $zero, 20
ori $t2, $zero, 24
ori $t3, $zero, 12
ori $t4, $zero, 16
subu $t5, $t0, $t1
ori $t7, $zero, 20
ori $t8, $zero, 24
ori $t6, $t5, 13
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000014
260@00003008: $10 <= 00000018
300@0000300c: $11 <= 0000000c
340@00003010: $12 <= 00000010
380@00003014: $13 <= fffffff4
420@00003018: $15 <= 00000014
460@0000301c: $24 <= 00000018
500@00003020: $14 <= fffffffd

Ld_E_RS(ori)
ori $t0, $zero, 8
ori $t1, $zero, 20
ori $t2, $zero, 4
ori $t3, $zero, 12
ori $t4, $zero, 16
sw $t0, 0($t1)
lw $t5, 0($t1)
ori $t6, $t5, 13
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000014
260@00003008: $10 <= 00000004
300@0000300c: $11 <= 0000000c
340@00003010: $12 <= 00000010
340@00003014: *00000014 <= 00000008
420@00003018: $13 <= 00000008
500@0000301c: $14 <= 0000000d

Ld_M_RS(ori)
ori $t0, $zero, 8
ori $t1, $zero, 20
ori $t2, $zero, 4
ori $t3, $zero, 12
ori $t4, $zero, 16
sw $t0, 0($t1)
lw $t5, 0($t1)
ori $t7, $zero, 20
ori $t6, $t5, 13
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000014
260@00003008: $10 <= 00000004
300@0000300c: $11 <= 0000000c
340@00003010: $12 <= 00000010
340@00003014: *00000014 <= 00000008
420@00003018: $13 <= 00000008
460@0000301c: $15 <= 00000014
500@00003020: $14 <= 0000000d

Ld_W_RS(ori)
ori $t0, $zero, 8
ori $t1, $zero, 20
ori $t2, $zero, 4
ori $t3, $zero, 12
ori $t4, $zero, 16
sw $t0, 0($t1)
lw $t5, 0($t1)
ori $t7, $zero, 20
ori $t8, $zero, 24
ori $t6, $t5, 13
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000014
260@00003008: $10 <= 00000004
300@0000300c: $11 <= 0000000c
340@00003010: $12 <= 00000010
340@00003014: *00000014 <= 00000008
420@00003018: $13 <= 00000008
460@0000301c: $15 <= 00000014
500@00003020: $24 <= 00000018
540@00003024: $14 <= 0000000d

```
Jal_E_RS(ori)
ori $t0, $zero, 8
ori $t1, $zero, 20
jal change1
ori $t2, $ra, 8
ori $t3, $zero, 14
change1:
        ori $t4, $zero, 18
ori $t5, $zero, 22
ori $t6, $zero, 26
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000014
260@00003008: $31 <= 00003010
300@0000300c: $10 <= 00003018
340@00003014: $12 <= 00000012
380@00003018: $13 <= 00000016
420@0000301c: $14 <= 0000001a

Jal_M_RS(ori)
ori $t0, $zero, 8
ori $t1, $zero, 20
jal change1
ori $t2, $zero, 4
ori $t3, $zero, 14
change1:
        ori $t4, $ra, 18
ori $t5, $zero, 22
ori $t6, $zero, 26
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000014
260@00003008: $31 <= 00003010
300@0000300c: $10 <= 00000004
340@00003014: $12 <= 00003012
380@00003018: $13 <= 00000016
420@0000301c: $14 <= 0000001a


Jal_W_RS(ori)
ori $t0, $zero, 8
ori $t1, $zero, 20
jal change1
```

```
ori $t2, $zero, 4
ori $t3, $zero, 14
change1:
        ori $t7, $zero, 6
        ori $t4, $ra, 18
ori $t5, $zero, 22
ori $t6, $zero, 26
180@00003000: $ 8 <= 00000008
220@00003004: $ 9 <= 00000014
260@00003008: $31 <= 00003010
300@0000300c: $10 <= 00000004
340@00003014: $15 <= 00000006
380@00003018: $12 <= 00003012
420@0000301c: $13 <= 00000016
460@00003020: $14 <= 0000001a
```

Lw:

```
R_E/M/W_RS(lw)
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 9
ori $t3, $zero, 12
sw $t1, 0($t0)
sw $t2, 4($t0)
sw $t3, 8($t0)

occasion1:      #R_E_RS
        subu $t4, $t1, $t0
        lw $t5, 0($t4)

occasion2:      #R_M_RS
        subu $t5, $t3, $t0
        ori $zero, $zero, 5
        lw $t6, 0($t5)

occasion3:      #R_W_RS
        addu $t6, $t0, $t1
        ori $s0, $zero, 12
        ori $s1, $zero, 16
        lw $t7, 0($t6)
$ 8 <= 00000004
$ 9 <= 00000008
```

```
$10 <= 00000009
$11 <= 0000000c
*00000004 <= 00000008
*00000008 <= 00000009
*0000000c <= 0000000c
$12 <= 00000004
$13 <= 00000008
$13 <= 00000008
$14 <= 00000009
$14 <= 0000000c
$16 <= 0000000c
$17 <= 00000010
$15 <= 0000000c

I_E/M/W_RS(lw)
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 9
ori $t3, $zero, 12
sw $t1, 0($t0)
sw $t2, 4($t0)
sw $t3, 8($t0)

occasion1:      #R_E_RS
        ori $t4, $zero, 4
        lw $t5, 0($t4)

occasion2:      #R_M_RS
        ori $t5, $zero, 8
        ori $zero, $zero, 5
        lw $t6, 0($t5)

occasion3:      #R_W_RS
        ori $t6, $zero, 12
        ori $s0, $zero, 12
        ori $s1, $zero, 16
        lw $t7, 0($t6)
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 00000009
$11 <= 0000000c
*00000004 <= 00000008
*00000008 <= 00000009
```

```
*0000000c <= 0000000c
$12 <= 00000004
$13 <= 00000008
$13 <= 00000008
$14 <= 00000009
$14 <= 0000000c
$16 <= 0000000c
$17 <= 00000010
$15 <= 0000000c
Ld_E/M/W_RS(lw)
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 12
ori $t3, $zero, 16
sw $t0, 0($zero)
sw $t1, 0($t0)
sw $t2, 0($t1)
sw $t3, 4($t1)

 occasion1:     #ld_E_RS
          lw $t4, 0($t0)
          lw $t5, 0($t4)

 occasion2:     #ld_M_RS
          lw $t5, -4($t0)
          addu $zero, $zero, $t1
          lw $t6, 0($t5)

 occasion:      #ld_W_RS
          lw $t6, 4($t0)
          ori $s0, $zero, 1
          ori $s1, $zero, 2
          lw $t7, 0($t6)
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 0000000c
$11 <= 00000010
*00000000 <= 00000004
*00000004 <= 00000008
*00000008 <= 0000000c
*0000000c <= 00000010
$12 <= 00000008
$13 <= 0000000c
```

$13 <= 00000004
$14 <= 00000008
$14 <= 0000000c
$16 <= 00000001
$17 <= 00000002
$15 <= 00000010

由于 DM 的容量只有 4KB，因此 31 号寄存器中的值不可能作为 lw 指令中的寻址

Sw:

```
R_E/M/W_RS/RT(sw)
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 12
ori $t3, $zero, 16
ori $t4, $zero, 20

occasion1:      #R_E_RS
        addu $t5, $t0, $t1
        sw $t2, 0($t5)

occasion2:      #R_M_RS
        subu $t6, $t2, $t1
        ori $s0, $zero, 12
        sw $t3, 4($t6)

occasion3:      #R_W_RS
        subu $t7, $t4, $t0
        ori $s1, $zero, 4
        ori $s2, $zero, 8
        sw $t4, 0($t7)

occasion4:      #R_E_RT
        subu $t5, $t1, $t0
        sw $t5, 0($t0)

occasion5:      #R_M_RT
        subu $t5, $t2, $t0
        ori $s0, $zero, 2
        sw $t5, 4($t2)
```

```
occasion6:      #R_W_RT
        addu $t6, $t3, $t4
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        sw $t6, 0($t1)
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 0000000c
$11 <= 00000010
$12 <= 00000014
$13 <= 0000000c
*0000000c <= 0000000c
$14 <= 00000004
$16 <= 0000000c
*00000008 <= 00000010
$15 <= 00000010
$17 <= 00000004
$18 <= 00000008
*00000010 <= 00000014
$13 <= 00000004
*00000004 <= 00000004
$13 <= 00000008
$16 <= 00000002
*00000010 <= 00000008
$14 <= 00000024
$16 <= 00000001
$17 <= 00000002
*00000008 <= 00000024


I_E/M/W_RS/RT(sw)
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 12
ori $t3, $zero, 16
ori $t4, $zero, 20

occasion1:      #I_E_RS
        ori $t5, $zero, 12
        sw $t2, 0($t5)

occasion2:      #I_M_RS
        ori $t6, $zero, 4
```

```
        ori $s0, $zero, 12
        sw $t3, 4($t6)

occasion3:      #I_W_RS
        ori $t7, $zero, 16
        ori $s1, $zero, 4
        ori $s2, $zero, 8
        sw $t4, 0($t7)

occasion4:      #I_E_RT
        lui $t5, 3
        sw $t5, 0($t0)

occasion5:      #I_M_RT
        lui $t5, 1
        ori $s0, $zero, 2
        sw $t5, 4($t2)

occasion6:      #I_W_RT
        ori $zero, $zero, 9
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        sw $zero, 0($t1)
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 0000000c
$11 <= 00000010
$12 <= 00000014
$13 <= 0000000c
*0000000c <= 0000000c
$14 <= 00000004
$16 <= 0000000c
*00000008 <= 00000010
$15 <= 00000010
$17 <= 00000004
$18 <= 00000008
*00000010 <= 00000014
$13 <= 00030000
*00000004 <= 00030000
$13 <= 00010000
$16 <= 00000002
*00000010 <= 00010000
$16 <= 00000001
```

$17 <= 00000002
*00000008 <= 00000000

Ld_E/M/W_RS_RT(sw)

```
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 12
ori $t3, $zero, 16
ori $t4, $zero, 20
sw $t0, 0($zero)
sw $t1, 0($t0)
sw $t2, 4($t0)
sw $t3, 8($t0)
occasion1:      #ld_E_RS
        lw $t5, 0($t0)
        sw $t2, 0($t5)

occasion2:      #ld_M_RS
        lw $t6, 0($t0)
        ori $s0, $zero, 12
        sw $t3, 4($t6)

occasion3:      #ld_W_RS
        lw $t7, 4($t0)
        ori $s1, $zero, 4
        ori $s2, $zero, 8
        sw $t4, 0($t7)

occasion4:      #ld_E_RT
        lw $t5, 0($t2)
        sw $t5, 0($t0)

occasion5:      #ld_M_RT
        lw $t5, 4($t2)
        ori $s0, $zero, 2
        sw $t5, 4($t2)

occasion6:      #ld_W_RT
        lw $t6, 4($t0)
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        sw $t6, 0($t1)
```

$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 0000000c
$11 <= 00000010
$12 <= 00000014
*00000000 <= 00000004
*00000004 <= 00000008
*00000008 <= 0000000c
*0000000c <= 00000010
$13 <= 00000008
*00000008 <= 0000000c
$14 <= 00000008
$16 <= 0000000c
*0000000c <= 00000010
$15 <= 0000000c
$17 <= 00000004
$18 <= 00000008
*0000000c <= 00000014
$13 <= 00000014
*00000004 <= 00000014
$13 <= 00000000
$16 <= 00000002
*00000010 <= 00000000
$14 <= 0000000c
$16 <= 00000001
$17 <= 00000002
*00000008 <= 0000000c

(jal_sw 的情况)

Beq:

R_E/M/W_RS/RT(beq)(equal)

```
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 12
ori $t3, $zero, 16
subu $t4, $t0, $t1

occasion1:      #R_E_RS
        addu $t5, $t0, $t1
        beq $t5, $t2, change1
        ori $s0, $zero, 1
```

```
        ori $s1, $zero, 2
        change1:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion2:      #R_M_RS
        addu $t6, $t0, $t1
        ori $s0, $zero, 1
        beq $t6, $t2, change2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change2:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion3:      #R_W_RS
        addu $t7, $t0, $t1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        beq $t7, $t2, change3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change3:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion4:      #R_E_RT
        subu $t5, $t1, $t2
        beq $t4, $t5, change4
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change4:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion5:      #R_M_RT
        subu $t6, $t1, $t2
        ori $s0, $zero, 1
        beq $t4, $t6, change5
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change5:
                ori $s2, $zero, 3
```

```
                ori $s3, $zero, 4

occasion6:      #R_W_RT
        subu $t7, $t1, $t2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        beq $t4, $t7, change6
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change6:
                ori $s2, $zero, 3
                ori $s3, $zero, 4
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 0000000c
$11 <= 00000010
$12 <= fffffffc
$13 <= 0000000c
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004
$14 <= 0000000c
$16 <= 00000001
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004
$15 <= 0000000c
$16 <= 00000001
$17 <= 00000002
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004
$13 <= fffffffc
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004
$14 <= fffffffc
$16 <= 00000001
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004
$15 <= fffffffc
$16 <= 00000001
```

$17 <= 00000002
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004

R_E/M/W_RS/RT(beq)(unequal)

```
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 12
ori $t3, $zero, 16
subu $t4, $t0, $t1

occasion1:      #R_E_RS
        addu $t5, $t0, $t1
        beq $t5, $t3, change1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change1:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion2:      #R_M_RS
        addu $t6, $t0, $t1
        ori $s0, $zero, 1
        beq $t6, $t3, change2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change2:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion3:      #R_W_RS
        addu $t7, $t0, $t1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        beq $t7, $t3, change3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change3:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion4:      #R_E_RT
```

```
        subu $t5, $t1, $t2
        beq $t3, $t5, change4
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change4:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion5:      #R_M_RT
        subu $t6, $t1, $t2
        ori $s0, $zero, 1
        beq $t3, $t6, change5
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change5:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion6:      #R_W_RT
        subu $t7, $t1, $t2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        beq $t3, $t7, change6
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change6:
                ori $s2, $zero, 3
                ori $s3, $zero, 4
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 0000000c
$11 <= 00000010
$12 <= fffffffc
$13 <= 0000000c
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$19 <= 00000004
$14 <= 0000000c
$16 <= 00000001
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
```

```
$19 <= 00000004
$15 <= 0000000c
$16 <= 00000001
$17 <= 00000002
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$19 <= 00000004
$13 <= fffffffc
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$19 <= 00000004
$14 <= fffffffc
$16 <= 00000001
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$19 <= 00000004
$15 <= fffffffc
$16 <= 00000001
$17 <= 00000002
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$19 <= 00000004
```

I_E/M/W_RS/RT(beq)(equal)

```
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 12
ori $t3, $zero, 16
subu $t4, $t0, $t1

occasion1:      #I_E_RS
        ori $t5, $zero, 16
        beq $t5, $t3, change1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
```

```
        change1:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion2:      #I_M_RS
        ori $t6, $zero, 16
        ori $s0, $zero, 1
        beq $t6, $t3, change2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change2:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion3:      #I_W_RS
        ori $t7, $zero, 16
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        beq $t7, $t3, change3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change3:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion4:      #I_E_RT
        ori $t5, $zero, 8
        beq $t1, $t5, change4
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change4:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion5:      #I_M_RT
        ori $t6, $zero, 8
        ori $s0, $zero, 1
        beq $t1, $t6, change5
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change5:
                ori $s2, $zero, 3
                ori $s3, $zero, 4
```

```
occasion6:      #I_W_RT
        ori $t7, $zero, 8
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        beq $t1, $t7, change6
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change6:
                ori $s2, $zero, 3
                ori $s3, $zero, 4
```

```
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 0000000c
$11 <= 00000010
$12 <= fffffffc
$13 <= 00000010
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004
$14 <= 00000010
$16 <= 00000001
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004
$15 <= 00000010
$16 <= 00000001
$17 <= 00000002
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004
$13 <= 00000008
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004
$14 <= 00000008
$16 <= 00000001
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004
$15 <= 00000008
$16 <= 00000001
```

```
$17 <= 00000002
$16 <= 00000001
$18 <= 00000003
$19 <= 00000004

I_E/M/W_RS/RT(beq)(unequal)
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 12
ori $t3, $zero, 16
subu $t4, $t0, $t1

occasion1:      #I_E_RS
        ori $t5, $zero, 16
        beq $t5, $t2, change1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change1:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion2:      #I_M_RS
        ori $t6, $zero, 16
        ori $s0, $zero, 1
        beq $t6, $t2, change2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change2:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion3:      #I_W_RS
        ori $t7, $zero, 16
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        beq $t7, $t2, change3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change3:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion4:      #I_E_RT
```

```
        ori $t5, $zero, 8
        beq $t2, $t5, change4
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change4:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion5:      #I_M_RT
        ori $t6, $zero, 8
        ori $s0, $zero, 1
        beq $t2, $t6, change5
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change5:
                ori $s2, $zero, 3
                ori $s3, $zero, 4

occasion6:      #I_W_RT
        ori $t7, $zero, 8
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        beq $t2, $t7, change6
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change6:
                ori $s2, $zero, 3
                ori $s3, $zero, 4
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 0000000c
$11 <= 00000010
$12 <= fffffffc
$13 <= 00000010
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$19 <= 00000004
$14 <= 00000010
$16 <= 00000001
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
```

```
$19 <= 00000004
$15 <= 00000010
$16 <= 00000001
$17 <= 00000002
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$19 <= 00000004
$13 <= 00000008
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$19 <= 00000004
$14 <= 00000008
$16 <= 00000001
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$19 <= 00000004
$15 <= 00000008
$16 <= 00000001
$17 <= 00000002
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$19 <= 00000004
```

Ld_E/M/W_RS/RT(beq)(equal)

```
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 12
ori $s0, $zero, 1
sw $t0, 0($zero)
sw $t1, 4($zero)
sw $t2, 8($zero)
ori $s0, $zero, 1
ori $s1, $zero, 2

occasion1:      #ld_E_RS
        lw $t3, 0($t0)
        beq $t3, $t1, change1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
```

```
        change1:
                ori $s2, $zero, 2
                ori $s3, $zero, 3

occasion2:      #ld_M_RS
        lw $t4, 0($t0)
        ori $s2, $zero, 2
        beq $t4, $t1, change2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change2:
                ori $s2, $zero, 2
                ori $s3, $zero, 3

occasion3:      #ld_W_RS
        lw $t5, 0($t0)
        ori $s2, $zero, 2
        ori $s3, $zero, 3
        beq $t5, $t1, change3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change3:
                ori $s2, $zero, 2
                ori $s3, $zero, 3

occasion4:      #ld_E_RT
        lw $t6, 0($t0)
        beq $t6, $t1, change4
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change4:
                ori $s2, $zero, 2
                ori $s3, $zero, 3

occasion5:      #ld_M_RT
        lw $t7, 0($t0)
        ori $s2, $zero, 2
        beq $t7, $t1, change5
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change5:
                ori $s2, $zero, 2
                ori $s3, $zero, 3
```

```
occasion6:      #ld_W_RT
        lw $t8, 0($t0)
        ori $s2, $zero, 2
        ori $s3, $zero, 3
        beq $t8, $t1, change6
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change6:
                ori $s2, $zero, 2
                ori $s3, $zero, 3
```

```
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 0000000c
$16 <= 00000001
*00000000 <= 00000004
*00000004 <= 00000008
*00000008 <= 0000000c
$16 <= 00000001
$17 <= 00000002
$11 <= 00000008
$16 <= 00000001
$18 <= 00000002
$19 <= 00000003
$12 <= 00000008
$18 <= 00000002
$16 <= 00000001
$18 <= 00000002
$19 <= 00000003
$13 <= 00000008
$18 <= 00000002
$19 <= 00000003
$16 <= 00000001
$18 <= 00000002
$19 <= 00000003
$14 <= 00000008
$16 <= 00000001
$18 <= 00000002
$19 <= 00000003
$15 <= 00000008
$18 <= 00000002
$16 <= 00000001
```

```
$18 <= 00000002
$19 <= 00000003
$24 <= 00000008
$18 <= 00000002
$19 <= 00000003
$16 <= 00000001
$18 <= 00000002
$19 <= 00000003
```

Ld_E/M/W_RS/RT(beq)(unequal)

```
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 12
ori $s0, $zero, 1
sw $t0, 0($zero)
sw $t1, 4($zero)
sw $t2, 8($zero)
ori $s0, $zero, 1
ori $s1, $zero, 2

occasion1:      #ld_E_RS
        lw $t3, 0($t0)
        beq $t3, $t0, change1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change1:
                ori $s2, $zero, 2
                ori $s3, $zero, 3

occasion2:      #ld_M_RS
        lw $t4, 0($t0)
        ori $s2, $zero, 2
        beq $t4, $t0, change2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change2:
                ori $s2, $zero, 2
                ori $s3, $zero, 3

occasion3:      #ld_W_RS
        lw $t5, 0($t0)
        ori $s2, $zero, 2
        ori $s3, $zero, 3
```

```
        beq $t5, $t0, change3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change3:
                ori $s2, $zero, 2
                ori $s3, $zero, 3

occasion4:      #ld_E_RT
        lw $t6, 0($t0)
        beq $t0, $t6, change4
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change4:
                ori $s2, $zero, 2
                ori $s3, $zero, 3

occasion5:      #ld_M_RT
        lw $t7, 0($t0)
        ori $s2, $zero, 2
        beq $t0, $t7, change5
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change5:
                ori $s2, $zero, 2
                ori $s3, $zero, 3

occasion6:      #ld_W_RT
        lw $t8, 0($t0)
        ori $s2, $zero, 2
        ori $s3, $zero, 3
        beq $t0, $t8, change6
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change6:
                ori $s2, $zero, 2
                ori $s3, $zero, 3
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 0000000c
$16 <= 00000001
*00000000 <= 00000004
*00000004 <= 00000008
*00000008 <= 0000000c
```

$16 <= 00000001
$17 <= 00000002
$11 <= 00000008
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$12 <= 00000008
$18 <= 00000002
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$13 <= 00000008
$18 <= 00000002
$19 <= 00000003
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$14 <= 00000008
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$15 <= 00000008
$18 <= 00000002
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$24 <= 00000008
$18 <= 00000002
$19 <= 00000003
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003

Jal_M/W_RS/RT(beq)(equal)
ori $t0, $zero, 4
ori $t1, $zero, 0x00003014
ori $t2, $zero, 0x00003034

```
ori $t3, $zero, 0x00003058
ori $t4, $zero, 0x00003078
occasion1:      #jal_M_RS
        jal change1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change1:
                beq $ra, $t1, change11
                ori $s0, $zero, 1
                ori $s1, $zero, 2
                change11:
                        ori $s2, $zero, 2
                        ori $s3, $zero, 3
occasion2:      #jal_W_RS
        jal change2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change2:
                ori $s2, $zero, 2
                beq $ra, $t2, change21
                ori $s0, $zero, 1
                ori $s1, $zero, 2
                change21:
                        ori $s2, $zero, 2
                        ori $s3, $zero, 3

occasion3:      #jal_M_RT
        jal change3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change3:
                beq $t3, $ra, change31
                ori $s0, $zero, 1
                ori $s1, $zero, 2
                change31:
                        ori $s2, $zero, 2
                        ori $s3, $zero, 3
occasion4:      #jal_W_RT
        jal change4
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change4:
                ori $s2, $zero, 2
```

```
                beq $t4, $ra, change41
                ori $s0, $zero, 1
                ori $s1, $zero, 2
                change41:
                        ori $s2, $zero, 2
                        ori $s3, $zero, 3
$ 8 <= 00000004
$ 9 <= 00003014
$10 <= 00003034
$11 <= 00003058
$12 <= 00003078
$31 <= 0000301c
$16 <= 00000001
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$31 <= 0000303c
$16 <= 00000001
$18 <= 00000002
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$31 <= 00003060
$16 <= 00000001
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$31 <= 00003080
$16 <= 00000001
$18 <= 00000002
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003

Jal_M/W_RS/RT(beq)(unequal)
ori $t0, $zero, 4
ori $t1, $zero, 0x00003010
ori $t2, $zero, 0x00003030
ori $t3, $zero, 0x00003050
```

```
ori $t4, $zero, 0x00003070
occasion1:      #jal_M_RS
        jal change1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change1:
                beq $ra, $t1, change11
                ori $s0, $zero, 1
                ori $s1, $zero, 2
                change11:
                        ori $s2, $zero, 2
                        ori $s3, $zero, 3
occasion2:      #jal_W_RS
        jal change2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change2:
                ori $s2, $zero, 2
                beq $ra, $t2, change21
                ori $s0, $zero, 1
                ori $s1, $zero, 2
                change21:
                        ori $s2, $zero, 2
                        ori $s3, $zero, 3

occasion3:      #jal_M_RT
        jal change3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change3:
                beq $t3, $ra, change31
                ori $s0, $zero, 1
                ori $s1, $zero, 2
                change31:
                        ori $s2, $zero, 2
                        ori $s3, $zero, 3
occasion4:      #jal_W_RT
        jal change4
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        change4:
                ori $s2, $zero, 2
                beq $t4, $ra, change41
```

```
                ori $s0, $zero, 1
                ori $s1, $zero, 2
                change41:
                        ori $s2, $zero, 2
                        ori $s3, $zero, 3
```

```
$ 8 <= 00000004
$ 9 <= 00003010
$10 <= 00003030
$11 <= 00003050
$12 <= 00003070
$31 <= 0000301c
$16 <= 00000001
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$31 <= 0000303c
$16 <= 00000001
$18 <= 00000002
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$31 <= 00003060
$16 <= 00000001
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$31 <= 00003080
$16 <= 00000001
$18 <= 00000002
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
```

```
Jal_M_RS(jr)
ori $t0, $zero, 4
ori $t1, $zero, 0x00003010
ori $t2, $zero, 0x00003030
```

```
ori $t3, $zero, 0x00003050
ori $t4, $zero, 0x00003070
occasion1:      #jal_M_RS
        jal change1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        ori $s2, $zero, 2
        ori $s3, $zero, 3
        change1:
                jr $ra
                ori $s0, $zero, 1
                ori $s1, $zero, 2
$ 8 <= 00000004
$ 9 <= 00003010
$10 <= 00003030
$11 <= 00003050
$12 <= 00003070
$31 <= 0000301c
$16 <= 00000001
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$16 <= 00000001

Jal_W_RS(jr)
ori $t0, $zero, 4
ori $t1, $zero, 0x00003010
ori $t2, $zero, 0x00003030
ori $t3, $zero, 0x00003050
ori $t4, $zero, 0x00003070
occasion1:      #jal_W_RS
        jal change1
        ori $s0, $zero, 1
        ori $s1, $zero, 2
```

```
        ori $s2, $zero, 2
        ori $s3, $zero, 3
        change1:
                ori $s3, $zero, 3
                jr $ra
                ori $s0, $zero, 1
                ori $s1, $zero, 2
$ 8 <= 00000004
$ 9 <= 00003010
$10 <= 00003030
$11 <= 00003050
$12 <= 00003070
$31 <= 0000301c
$16 <= 00000001
$19 <= 00000003
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$19 <= 00000003
$16 <= 00000001
$17 <= 00000002
$18 <= 00000002
$19 <= 00000003
$19 <= 00000003

ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 0x00003000
ori $s0, $zero, 1
ori $s1, $zero, 2
ori $s2, $zero, 3
occasion1:      #R_E_RS
        addu $t3, $t0, $t2
        jr $t3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        ori $s2, $zero, 3

$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
```

$17 <= 00000002
$18 <= 00000003
$11 <= 00003004
$16 <= 00000001
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003004
$16 <= 00000001
$ 9 <= 00000008
$10 <= 00003000

R_M_RS(jr)

```
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 0x00003000
ori $s0, $zero, 1
ori $s1, $zero, 2
ori $s2, $zero, 3
occasion1:      #R_M_RS
        addu $t3, $t0, $t2
        ori $s0, $zero, 1
        jr $t3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        ori $s2, $zero, 3
```

$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003004
$16 <= 00000001
$16 <= 00000001
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003004

```
$16 <= 00000001
$16 <= 00000001
$ 9 <= 00000008
$10 <= 00003000

R_W_RS(jr)
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 0x00003000
ori $s0, $zero, 1
ori $s1, $zero, 2
ori $s2, $zero, 3
occasion1:      #R_W_RS
        addu $t3, $t0, $t2
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        jr $t3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        ori $s2, $zero, 3
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003004
$16 <= 00000001
$17 <= 00000002
$16 <= 00000001
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003004
$16 <= 00000001
$17 <= 00000002
$16 <= 00000001

I_E_RS(jr)
ori $t0, $zero, 4
ori $t1, $zero, 8
```

```
ori $t2, $zero, 0x00003000
ori $s0, $zero, 1
ori $s1, $zero, 2
ori $s2, $zero, 3
occasion1:      #R_E_RS
        ori $t3, $t2, 0
        jr $t3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        ori $s2, $zero, 3
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003000
$16 <= 00000001
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003000
$16 <= 00000001
$ 8 <= 00000004
```

```
I_M_RS(jr)
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 0x00003000
ori $s0, $zero, 1
ori $s1, $zero, 2
ori $s2, $zero, 3
occasion1:      #R_M_RS
        ori $t3, $t2, 0
        ori $s0, $zero, 1
        jr $t3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        ori $s2, $zero, 3
```

```
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003000
$16 <= 00000001
$16 <= 00000001
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003000
$16 <= 00000001
$16 <= 00000001
$ 8 <= 00000004
```

I_W_RS(jr)

```
ori $t0, $zero, 4
ori $t1, $zero, 8
ori $t2, $zero, 0x00003000
ori $s0, $zero, 1
ori $s1, $zero, 2
ori $s2, $zero, 3
occasion1:      #R_W_RS
        ori $t3, $t2, 0
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        jr $t3
        ori $s0, $zero, 1
        ori $s1, $zero, 2
        ori $s2, $zero, 3
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003000
$16 <= 00000001
```

```
$17 <= 00000002
$16 <= 00000001
$ 8 <= 00000004
$ 9 <= 00000008
$10 <= 00003000
$16 <= 00000001
$17 <= 00000002
$18 <= 00000003
$11 <= 00003000
$16 <= 00000001
$17 <= 00000002
```

```
Ld_E_RS(jr)
ori $t0, $zero, 0X00003004
ori $t1, $zero, 0X00003008
ori $s0, $zero, 0
ori $s0, $zero, 0
sw $t0, 0($zero)
sw $t1, 4($zero)
ori $s0, $zero, 0
ori $s0, $zero, 0
occasion1:      #ld_E_RS
        lw $t2, 0($zero)
        jr $t2
        ori $s0, $zero, 0
        ori $s0, $zero, 0
        ori $s0, $zero, 0
$ 8 <= 00003004
$ 9 <= 00003008
$16 <= 00000000
$16 <= 00000000
*00000000 <= 00003004
*00000004 <= 00003008
$16 <= 00000000
$16 <= 00000000
$10 <= 00003004
$16 <= 00000000
$ 9 <= 00003008
$16 <= 00000000
$16 <= 00000000
```

```
*00000000 <= 00003004
*00000004 <= 00003008
$16 <= 00000000
$16 <= 00000000
$10 <= 00003004

ld_M_RS(jr)
ori $t0, $zero, 0X00003004
ori $t1, $zero, 0X00003008
ori $s0, $zero, 0
ori $s0, $zero, 0
sw $t0, 0($zero)
sw $t1, 4($zero)
ori $s0, $zero, 0
ori $s0, $zero, 0
occasion1:      #ld_M_RS
        lw $t2, 0($zero)
        ori $s0, $zero, 0
        jr $t2
        ori $s0, $zero, 0
        ori $s0, $zero, 0
        ori $s0, $zero, 0
$ 8 <= 00003004
$ 9 <= 00003008
$16 <= 00000000
$16 <= 00000000
*00000000 <= 00003004
*00000004 <= 00003008
$16 <= 00000000
$16 <= 00000000
$10 <= 00003004
$16 <= 00000000
$16 <= 00000000
$ 9 <= 00003008
$16 <= 00000000
$16 <= 00000000
*00000000 <= 00003004
*00000004 <= 00003008
$16 <= 00000000
$16 <= 00000000
$10 <= 00003004

ld_W_RS(jr)
```

```
ori $t0, $zero, 0X00003004
ori $t1, $zero, 0X00003008
ori $s0, $zero, 0
ori $s0, $zero, 0
sw $t0, 0($zero)
sw $t1, 4($zero)
ori $s0, $zero, 0
ori $s0, $zero, 0
occasion1:      #ld_W_RS
        lw $t2, 0($zero)
        ori $s0, $zero, 0
        ori $s0, $zero, 0
        jr $t2
        ori $s0, $zero, 0
        ori $s0, $zero, 0
        ori $s0, $zero, 0
$ 8 <= 00003004
$ 9 <= 00003008
$16 <= 00000000
$16 <= 00000000
*00000000 <= 00003004
*00000004 <= 00003008
$16 <= 00000000
$16 <= 00000000
$10 <= 00003004
$16 <= 00000000
$16 <= 00000000
$16 <= 00000000
$ 9 <= 00003008
$16 <= 00000000
$16 <= 00000000
*00000000 <= 00003004
*00000004 <= 00003008
$16 <= 00000000
$16 <= 00000000
$10 <= 00003004
```